ARMY RESEARCH LABORATORY

Software Tools for Measuring and Calculating Electromagnetic Shielding Effectiveness

by Neal Tesny

ARL-TR-3645 **September 2005**

Army Research Laboratory
Adelphi, MD 20783-1197

ARL-TR-3645 **September 2005**

Software Tools for Measuring and Calculating Electromagnetic Shielding Effectiveness

Neal Tesny
Sensors and Electron Devices Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED (From - To)**
September 2005	Final	April to October 2004

4. TITLE AND SUBTITLE
Software Tools for Measuring and Calculating Electromagnetic Shielding Effectiveness

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Neal Tesny

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: AMSRD-ARL-SE-DE
2800 Powder Mill Road
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-3645

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD 20783-1197

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The evaluation and the analysis of high-altitude electromagnetic pulse response of shielded enclosures require the availability of software tools able to acquire data and calculate shielding effectiveness. This report describes new software tools for measuring and calculating shielding effectiveness. The tools are written in MATLAB[1] in graphical user interface format and are unique in that they allow for real-time analysis and plotting of the results in graphical form. Sample output is also presented.

[1]MATLAB is a registered trademark of the MathWorks.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Neal Tesny
Unclassified	Unclassified	Unclassified	UL	46	**19b. TELEPHONE NUMBER** *(Include area code)* (301) 394-5559

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

Declaration

This is to certify that the software is the sole work of the author, with exception of minor general subroutines written for general use. The writer of these subprograms is mentioned in the acknowledgments.

Acknowledgments

I would like to acknowledge Dr. Marc Litz of the Army Research Laboratories Sensors Directorate for the creation of minor, general use subprograms that I employed in writing the software, particularly the subroutines “ascread” and “ascwrite” for file input/output.

I would also like to thank the Strategic Hardening Team of ARL for their painstaking “beta” testing of the software and recommendations for improvement.

1. Introduction

The determination of whether command, control, communications, computer, and intelligence (C4I) facilities provide adequate HEMP protection requires that extensive shielding effectiveness measurements of these facilities be conducted. The shielding effectiveness characteristics of a facility are obtained from calculations using large amounts of test data generated by measurements. When these measurements were taken, it became apparent that the ability to calculate and display the shielding effectiveness at a given test point quickly would significantly enhance and accelerate the determination of HEMP protection provided by the facility. To that end, software tools have been written to achieve that ability. The software tools consist of two parts: data acquisition and data analysis. Measured and processed data are displayed graphically via the use of graphical user interface (GUI). The code was written using MATLAB[1] (*1*).

2. Background

2.1 Definition of Shielding Effectiveness

Shielding effectiveness is "the relative capability of a shield to screen out undesirable electric and magnetic fields and plane waves. The measurement is the ratio of the signal received without the shield to the signal received inside the shield" (*2*). It is usually stated in decibels.

It is desired to be able to obtain measurements of shielding effectiveness for various enclosures in which electromagnetic leakage needs to be controlled. To do this, different measurement systems were developed over time. With the development of portable instrumentation and personal computers (PCs), a system eventually was developed that uses this state of the art.

The measurement system design was first successfully used during the U.S. Army's Heavy Terminal/Medium Terminal (HT/MT) upgrade program in the late 1990's (*3*, *4*). The reason for its development was threefold: first, to decrease the level of manpower needed to conduct the measurements, second, to accelerate the data acquisition process, and third, to obtain a higher quality level of data.

Early software versions consisted mostly of only that which could control the equipment and transfer the un-reduced data to a "floppy disk". Improvements came more recently in early 2000 and 2001 when improvements in the data acquisition software resulted in more flexible and efficient means to transfer data directly to modern, lap-top-driven media.

[1] MATLAB is a registered trademark of the MathWorks.

Additionally, newer antennas with broader effective bandwidths further reduced the number of data sweeps from three to two. At the end of 2003, the measurement systems' hardware evolved to the setup shown in figure 1. The data reduction and analysis functions, although having progressed to spread-sheet representations, still required labor-heavy analysis.

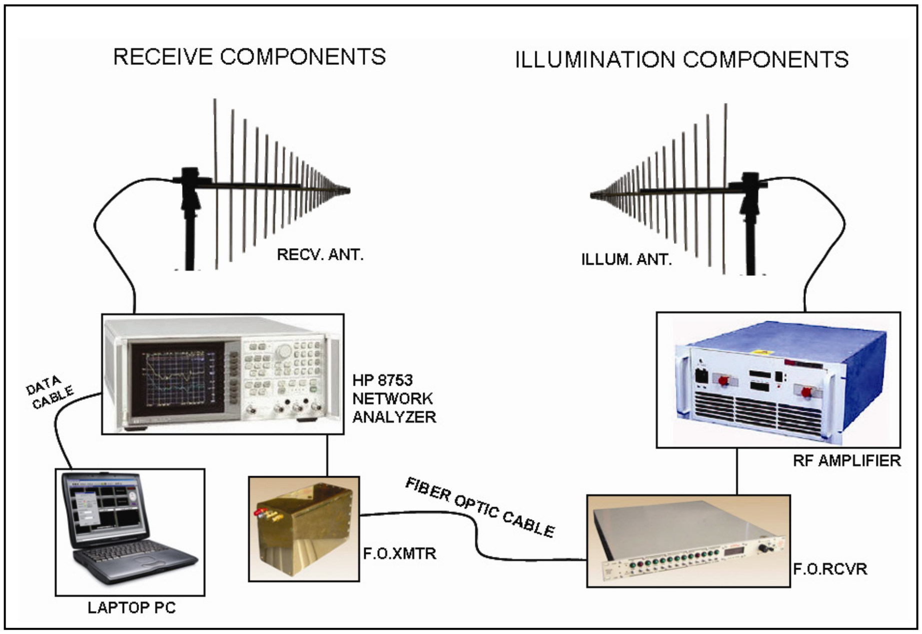

Figure 1. A typical measurement system.

2.2 Shielding Effectiveness Measurements

2.2.1 Procedure and Setup

SE measurements are usually taken in the following manner. A source sends a transmitted signal through an antenna, which is received through another antenna to a recording device. The two antennas are placed in such a way as to measure how the transmitted signal propagates through a medium, e.g., a radio frequency (RF) shield. An amplifying device is usually needed to boost the signal before it is transmitted in order to achieve adequate dynamic range at the receiver. The recording device is usually a very sensitive RF receiver, such as a spectrum analyzer or network analyzer. The receiver is typically connected to a portable PC for digital storage of the measured signal.

A setup of a typical measurement system is shown in figure 1. A network analyzer that provides its own source and can generate a large number of frequencies quickly is used. The output of the network analyzer (O) is split so that a reference signal (R) can be provided to compare with the signal received by the receive antenna (A). For a given frequency and polarization, the difference in intensity between the received signal and the reference signal, called A/R, is a measure of the signal loss at that frequency because of the path between the transmission antenna

and the reception antenna plus additional losses and gains because of the various components in the transmitting and receiving circuits, including cables, the signal splitter, fiber-optic equipment, amplifier and antennas. The shielding effectiveness at a given frequency is the difference between this value and the value at that frequency obtained when the system was calibrated, i.e., when there were no shelter walls between the transmission antenna and reception antenna. This is the case because the additional losses and gains just described are assumed to be the same for the calibration measurement and the shielding effectiveness measurement. Data recorded by the network analyzer are downloaded to a computer.

3. Data Acquisition Software Tools

Software was written to retrieve data from the network analyzer to PC by means of a general purpose interface bus (GPIB). These tools allow digital storage of data in a timely and user-friendly manner. A separate program tool exists for each network analyzer.

3.1 Guide to Using an Acquisition Tool

The operation of this software tool involves the following:

1. Evoking the program. In the MATLAB command window, type either "na8753menu" or "na3577menu" to start the program corresponding to which network analyzer is being used. The GUI menu then appears (see figure 2).

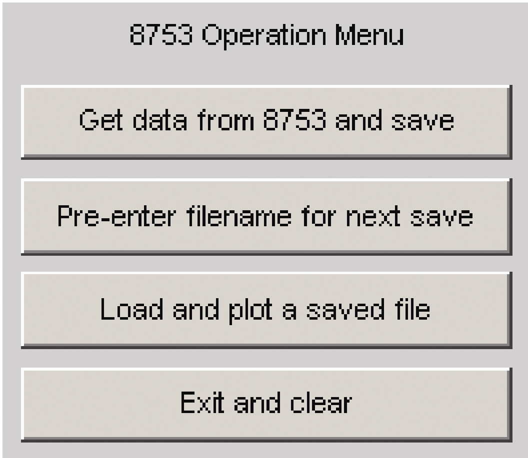

Figure 2. Data acquisition menu.

2. The user can now select from four options:

    a. Acquire data from network analyzer. When this option is selected, the data in the network analyzer are retrieved into the PC. The user is then prompted to enter a file name under which the data will be stored. After the user enters a file name, the data are then stored in an appropriate data file on the PC and are plotted graphically on screen.

    b. Pre-enter filename for next save. This option allows the user to enter the file name before the data are actually retrieved from the network analyzer. This allows maximum use of time during the measurements.

    c. Retrieve a saved file. This allows the user to recall data from a previously saved data file. The data are then plotted on screen.

    d. Exit program. The program is terminated and variables are cleared. It is necessary to do this when one is changing network analyzers because they use different GPIB variables.

---

4. SE Analysis

4.1 Types of Measurements Needed

In order to obtain SE, four types of measurements are taken: (1) calibration, or reference, (2) leakage or measurement of signal through shield, (3) ambient noise, and (4) calibration of attenuating or signal-conditioning devices. These are described as follow.

1. Calibration measurements are taken with the antennas placed at a pre-set distance with free space, i.e., nothing between them. The signal collected is known as the “reference” or “ealibration” and is used as a basis for calculating the SE.

2. The leakage measurement is the “in situ” measurement, i.e., the actual measurement of signal leaking through the RF shield. It is typically taken with the transmitting antenna placed outside the RF shield and the receiving antenna placed inside the RF shield. The total distance between the antennas is the same as was used for the calibration measurement.

3. An ambient noise measurement is made at the location where the leakage measurement is made. Ambient noise is the background signal that is recorded when the RF source is not transmitting. This gives a recording of the lowest possible signal which can be measured by the recording instruments.

4. A measurement of signal-conditioning devices is taken in order to accurately characterize their response. This is done if it is necessary to use these devices during the calibration or leakage measurements (e.g., because of the high amplitude of the signals recorded during the calibration measurement, it is often necessary to reduce their amplitude with external RF attenuators).

4.2 SE Calculations

SE is calculated when the calibration measurement at a given frequency is divided by the leak-through measurement at that frequency (or subtraction if they are expressed in decibels). Any attenuators or signal-conditioning devices must also be accounted for at the given frequency. This is expressed in the following equation for data expressed in decibels:

$$SE_{dB} = (Calibration_{dB} + Attenuation_{dB}) - Leakage_{dB}$$

in which

Calibration is the signal level recorded during the calibration process,

Attenuation is the response of the signal-conditioning devices, where attenuation is expressed as a positive number, and

Leakage is the signal recorded in the leak-through measurement.

4.3 Dynamic Range Calculations

In order to ensure meaningful results, the measurement system needs to be checked for proper dynamic range (DR). DR is the widest range of measurement that could be detected by the measurement system. We determine DR by measuring the ambient noise at the location where the leak-through measurement is made.

Similar to the SE calculation, DR at a given frequency is calculated when the calibration measurement is divided by the ambient noise measurement at that frequency (or subtraction if they are expressed in decibels). This is expressed in the following equation for data in decibels:

$$DR_{dB} = (Calibration_{dB} + Attenuation_{dB}) - Ambient\_Noise_{dB}$$

In which

Calibration is the signal level recorded during the calibration process,

Attenuation is the response of the signal-conditioning devices, where attenuation is expressed as a positive number, and

Ambient_Noise is the signal recorded in the ambient noise measurement.

5. SE Calculation Tool

5.1 Graphical User Interface (GUI)

The software uses a GUI to graphically load, calculate, and display all of the data. The GUI with data loaded is shown in figure 3.

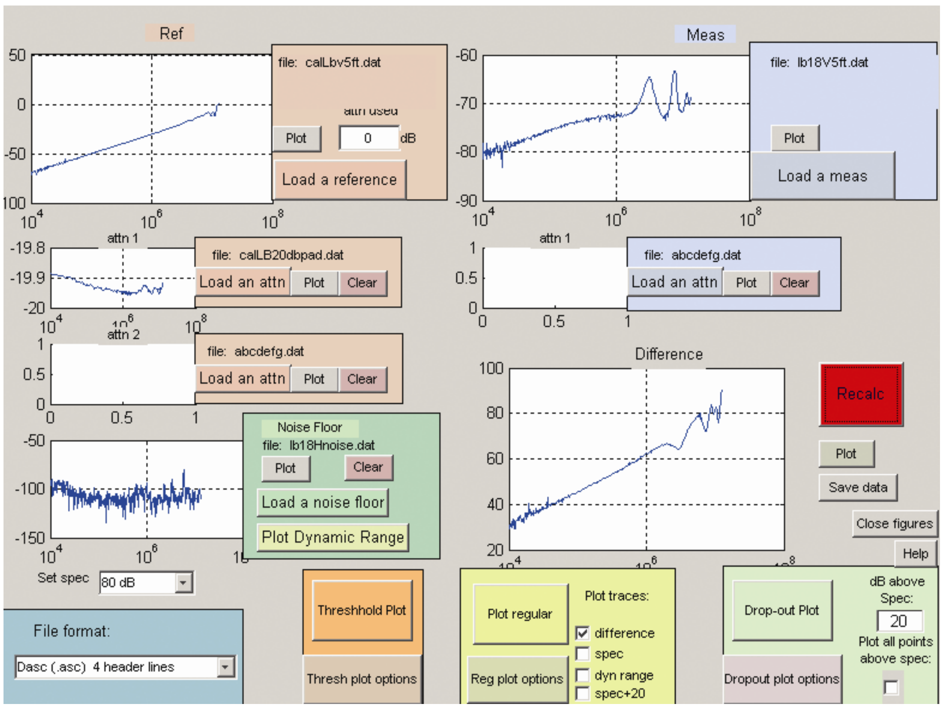

Figure 3. GUI panel.

The GUI makes the process of scanning the raw measurement data and calculating and displaying the results very easy, and it enables the user to immediately view the data plotted graphically. In this way, the user sees the final results almost immediately and can readily detect any anomalies or problems with the data. In addition, there is an option for the user to save the SE values to a file for later viewing.

5.2 Types of Plots

The GUI enables the user to generate four different types of plots for displaying data: regular plots, overlay plots, threshold plots, and drop-out plots. An example of each is shown in appendix A. There are many options that can be controlled by the user to change the appearance of the data plots. Some of these include line weight, line type, line color below threshold, line color above threshold, and options to change the appearance of lines, fonts, and other plot

characteristics. Default values exist for all the changeable options. These options are further discussed in section 6.2.

5.2.1 Regular Plots

Regular Plots are plots of a single waveform or signal and are plotted in a separate "pop-up" figure. These plots can be generated to display the calibration, leak-through, attenuation, ambient noise, dynamic range, or SE waveforms. An example of a regular plot is shown in figures A-1 and A-2.

5.2.2 Overlay Plots

Overlay Plots are plots of multiple waveforms. They can display SE, dynamic range, and SE specification on a single plot. An example is shown in figures A-3 and A-4.

5.2.3 Threshold Plots

Threshold Plots display data that fall below a SE specification or threshold differently than data that are above the threshold. This enables a user to clearly see when the SE is below a certain threshold. An example is shown in figures A-5 and A-6.

5.2.4 Drop-out Plots

Drop-out Plots are similar to threshold plots but only plot data if the dynamic range is above a certain threshold. That portion of the SE curve corresponding to the portion of the dynamic range curve that falls below the threshold is not plotted. The user can specify how much above the SE specification the dynamic range must be in order for it to be considered valid data. For example, if the minimum specification is 60 dB, entering a value of 20 dB will cause the SE to be plotted only for regions where the dynamic range is 80 dB. An example is shown in figures A-7 and A-8.

---

6. Guide to Using the SE Calculation Tool

In order for the program to be used, the data files must be labeled appropriately as to what data they represent. A sample minimum data set might include the following list of files (these file types were described in section 4.1):

"CAL1.DAT" – file containing the reference measurement,

"MEAS1.DAT" – file containing the leak-through measurement,

"NOISE1.DAT" – file containing the ambient noise measurement, and

"ATTN1.DAT" – file containing the measurement of the attenuator insertion loss.

6.1 Evoking the Program

In the MATLAB command window, type SE4 to start the program. The GUI panel then appears.

6.2 Loading Data

The reference, attenuator, ambient noise, and leak-through data are loaded when the user clicks the corresponding "load" button with the computer mouse. A Windows[2]-type list of files appears and the user is asked to select the name of the file that contains the appropriate stored data. The data "curve" is then loaded into the corresponding window on the GUI panel. The buttons that load data into the appropriate windows are

- Load a reference
- Load a meas
- Load a noise floor
- Load an atten

The pull-down menu labeled "file format" is used to select the format of the files that store the data. The default value is "DASC (.asc) 4 header lines". Also, if it is desired to remove an attenuator or ambient noise data set, this is done when the user presses the button marked "clear" next to corresponding data set.

6.3 Calculating SE

The user clicks on the "Recalc" button to calculate SE and display it in the lower right-hand window. This must be done whenever new data are loaded into any of the data windows.

6.4 Plotting

To plot the data, the user clicks on the appropriate button to get any of the plot types described in section 5.2. The data are then plotted in an independently generated window. The buttons that plot the shielding effectiveness are

---

[2] Windows is a trademark of Microsoft.

- Drop-out plot
- Threshold plot
- Regular plot

For a “regular plot,” the traces that can be plotted are selected by the check boxes under the heading “Plot traces” which is near the “Plot regular” button in the GUI panel.

For a drop-out plot, the user entry “dB above Spec” on the GUI panel is used to set the level where the dynamic range must be in order for the data points to be plotted, specified in decibels. This is further described in section 5.2.4. The check box labeled “Plot all points above spec” allows each data point that falls above the specification to be plotted, regardless of the dynamic range level corresponding to that data point.

In addition to plotting the final shielding effectiveness, each data trace that was loaded into the GUI can be plotted in an independent window when the user presses the “Plot” button next to its display in the GUI panel.

Plot options exist for each of the plots of shielding effectiveness. These are listed in section 6.2. The buttons that control plotting options are

- Thresh plot options
- Reg plot options
- Drop-out plot options

6.5 Other Controls

In addition to these controls, there are other buttons and controls that can be used to adjust, save, and remove plots. These are

- Set spec: This pull-down menu is used to set the minimum acceptable shielding effectiveness for the threshold and drop-out plots. Settable values are 60 dB and 80 dB.
- Close figures: This button removes all of the independently generated plot windows that have been generated.
- Save data: This button allows the user to save the shielding effectiveness in a separate data file. It produces a pop-up window that asks for a name of the file that will be saved and its location.

6.6 Plot Options

Many options exist for plotting the data. These are controlled through the GUI when the user clicks on the “Plot Options” buttons for the appropriate plots. The options include

- SE Line weight
- SE Line type
- SE Line color above threshold
- SE Line color below threshold
- SE data marker type above threshold
- SE data marker type below threshold
- SE data marker color above threshold
- SE data marker color below threshold
- Threshold Line weight
- Threshold Line color
- Threshold Line type
- Logarithmic or linear x-axis
- Plot grid on or off
- Plot title font size
- Plot x- and y-label font size
- Plot tic font size

7. Conclusions

An efficient, easy-to-use GUI for performing shielding effectiveness calculations has been demonstrated.

A new and novel software package has been developed for shielding effectiveness measurements and calculations. It provides easy-to-use data acquisition from state-of-the-art RF instruments and provides real-time analysis of data collected. The novel features include a user-friendly GUI that provides graphical output as opposed to only numerical, which allows immediate detection of failure points and frequencies where shielding effectiveness is deficient along with the exact amount of failure.

Abbreviations

RF	Radio Frequency
SE	Shielding Effectiveness
PC	Personal Computer
EM	Electromagnetic
DR	Dynamic Range
GUI (pronounced “goo-ee”)	Graphical User Interface
I/O	Input/Output
dB	decibels
GPIB	General Purpose Interface Bus

References

1. MATLAB, The MathWorks, Inc., Natick, MA, 1984-2004.

2. From Technical Marketing Specialists' web page, EMI & RFI glossary, August 2004, http://www.tmssales.com/Links/EMI-RFI%20Glossary%20and%20Acronyms.htm.

3. Atkinson, R. *Status Report: High-Altitude Electromagnetic Pulse Heavy Terminal Modification Support Efforts*; ARL-TR-1372; U.S. Army Research Laboratory: Adelphi, MD, December 1997, SECRET.

4. MIL-STD-188-125-1, *Department of Defense Interface Standard: High-Altitude Electromagnetic Pulse (HEMP) Protection For Ground-Based C4I Facilities Performing Critical, Time-Urgent Missions, Part 1: Fixed Facilities*. 17 July 1998.

Appendix A. Sample Output.

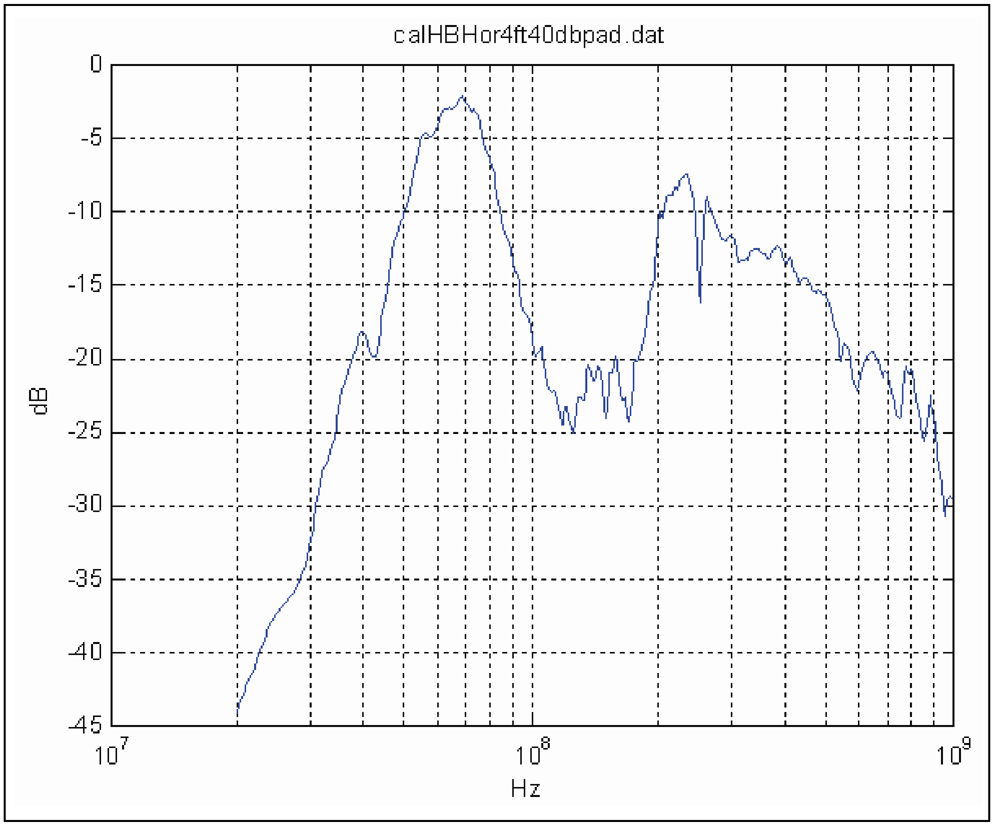

Figure A-1. Sample regular plot 1. (A calibration plot is shown.)

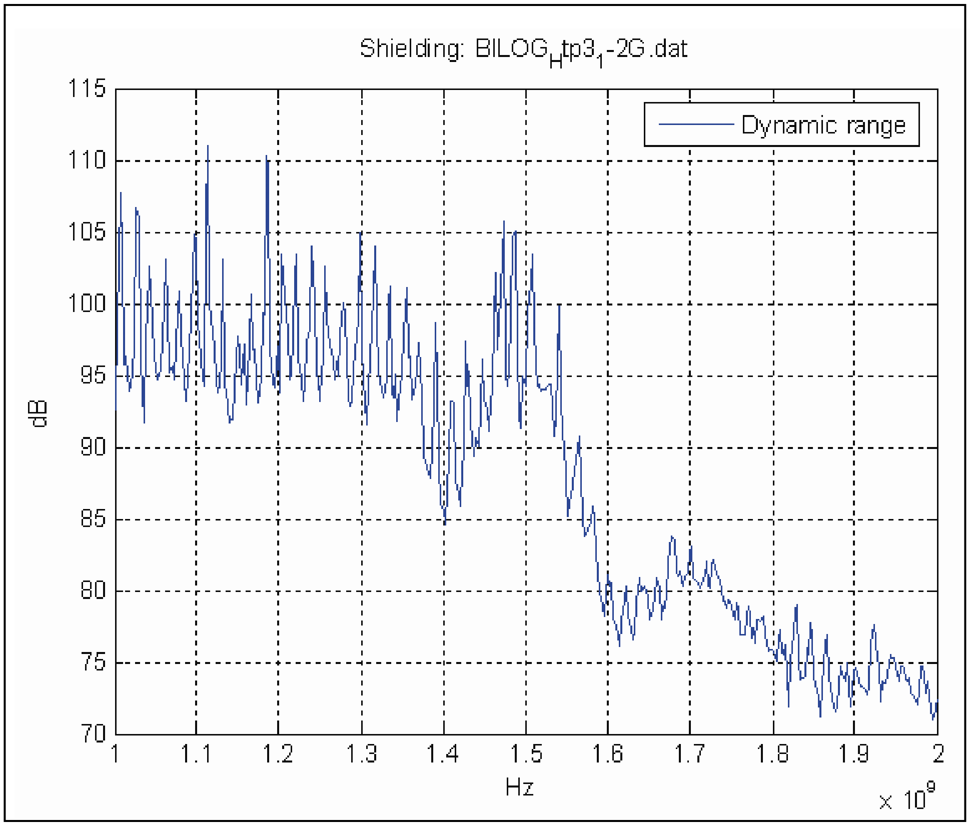

Figure A-2. Sample regular plot 2. (A dynamic range plot is shown.)

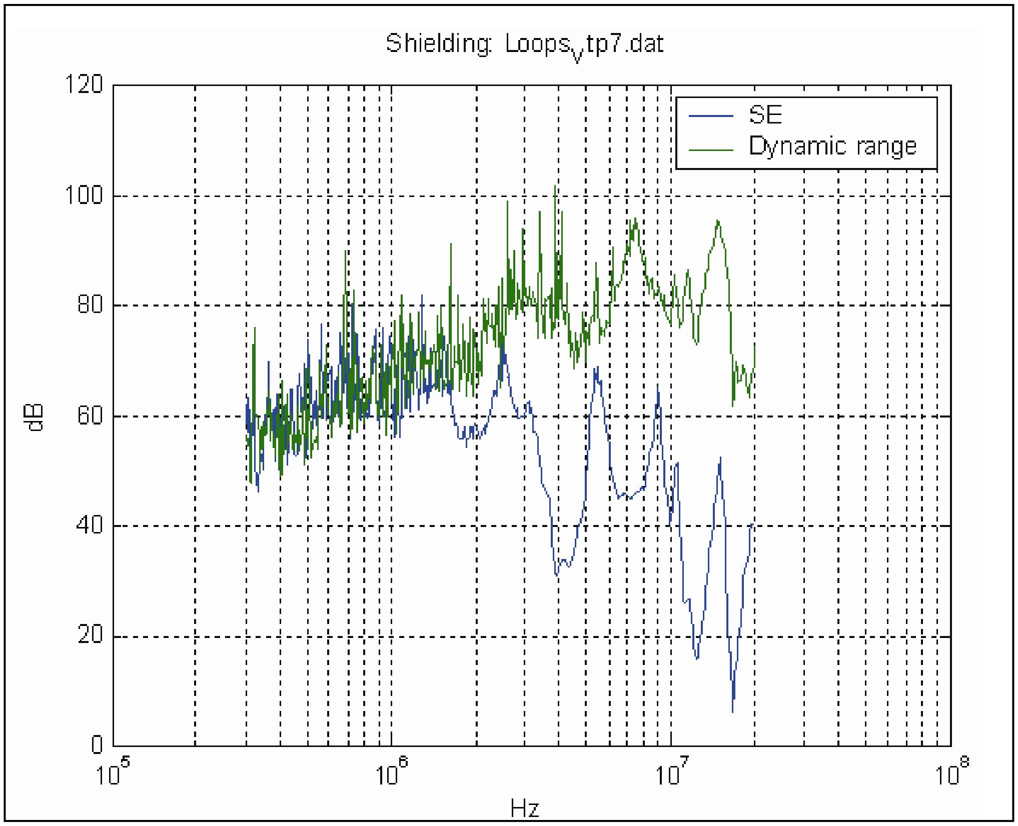

Figure A-3. Sample overlay plot 1. (The shielding effectiveness is shown in blue and the dynamic range in green.)

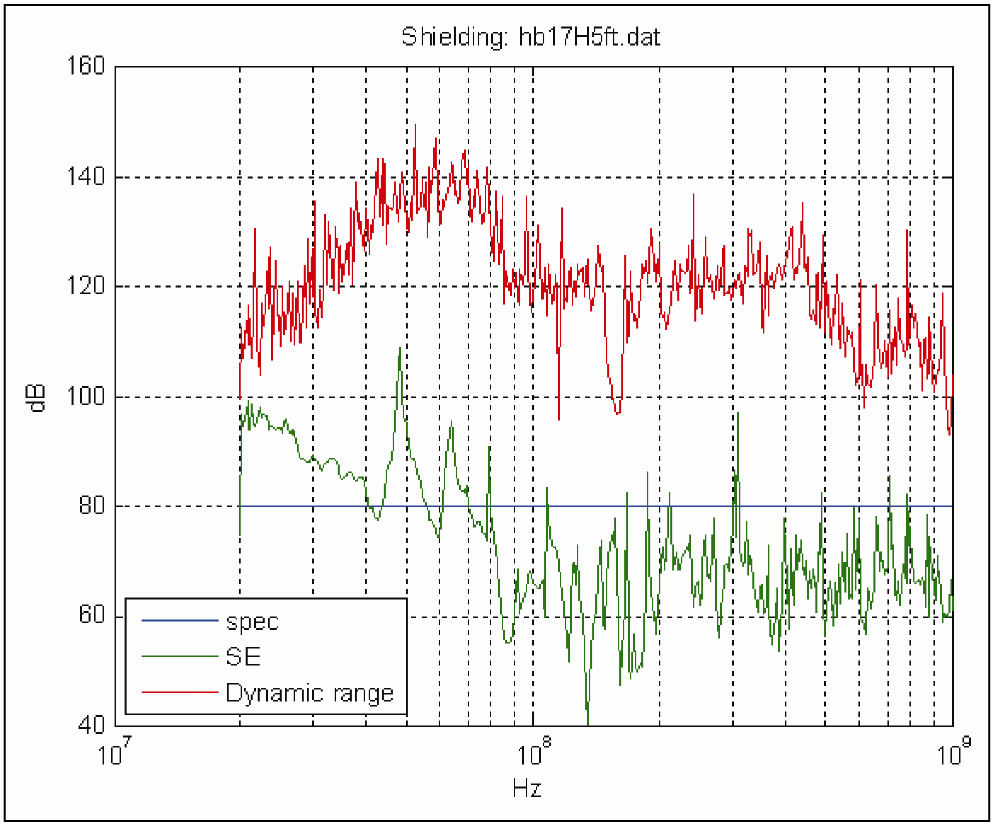

Figure A-4. Sample overlay plot 2. (The shielding effectiveness is shown in green, the criteria in blue, and the dynamic range in red.)

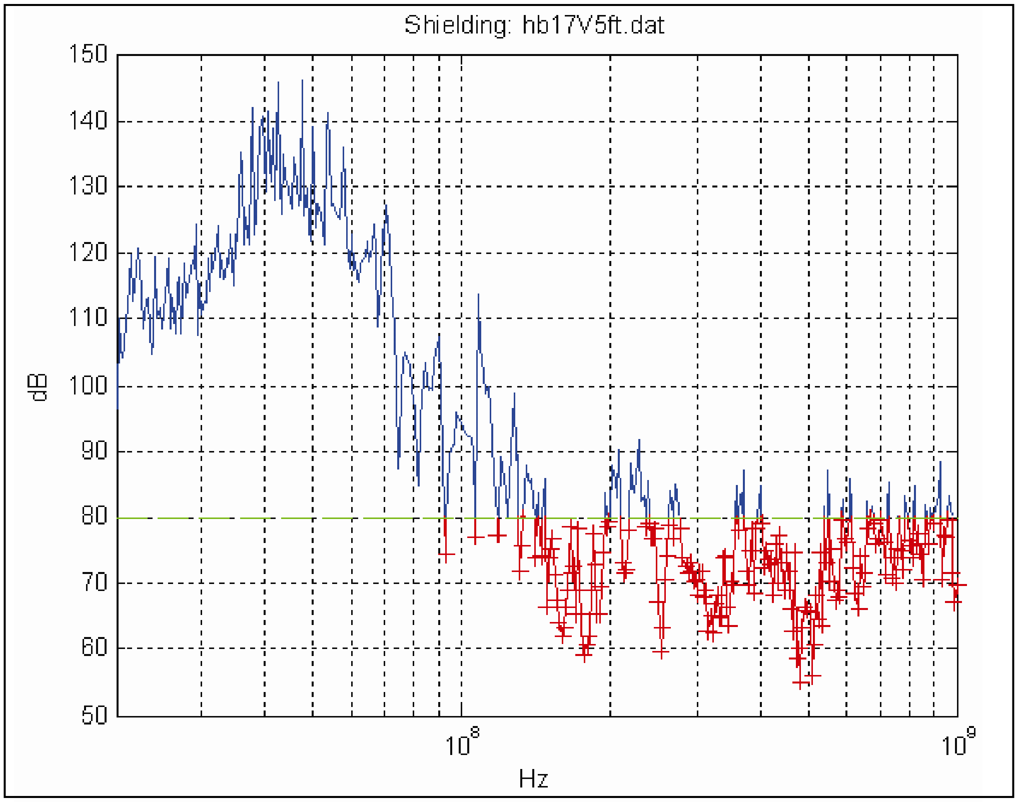

Figure A-5. Sample threshold plot showing region of satisfying criteria in blue and region of failing to meet criteria in red. (Criteria threshold is shown in green.)

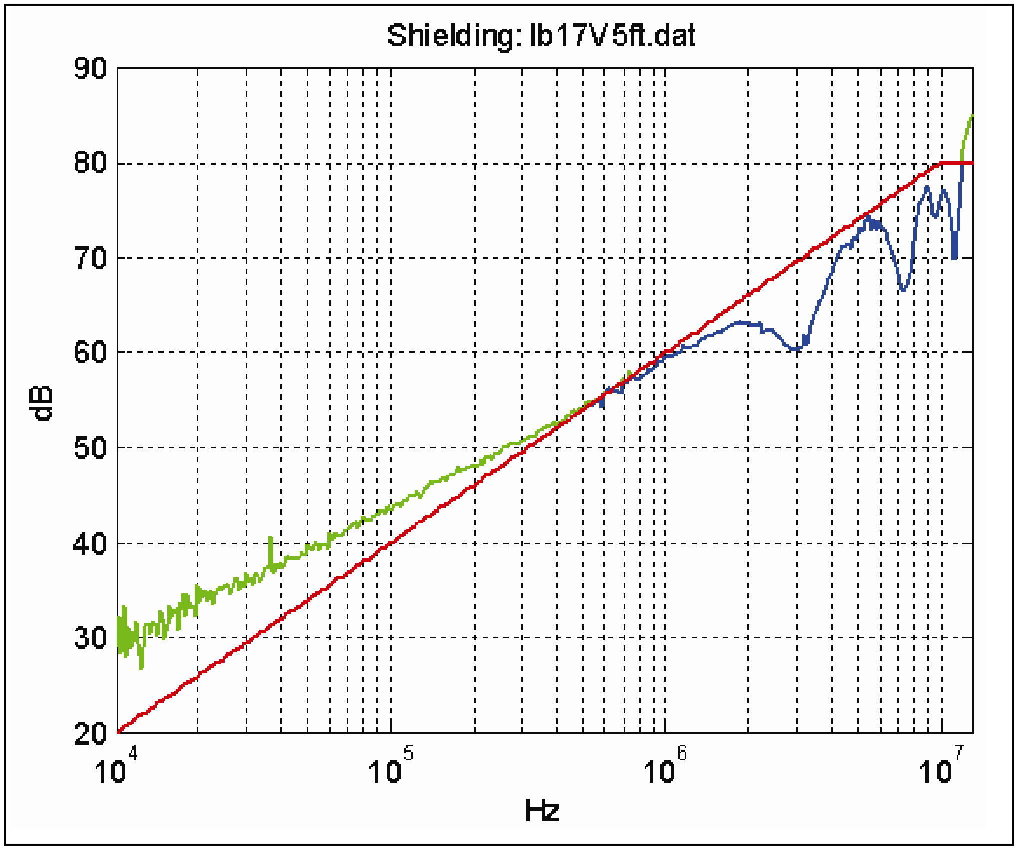

Figure A-6. Sample threshold plot. (A sloping criteria threshold is shown (red). The points falling above the criteria threshold are plotted in green, and the points falling below the criteria threshold are plotted in blue.)

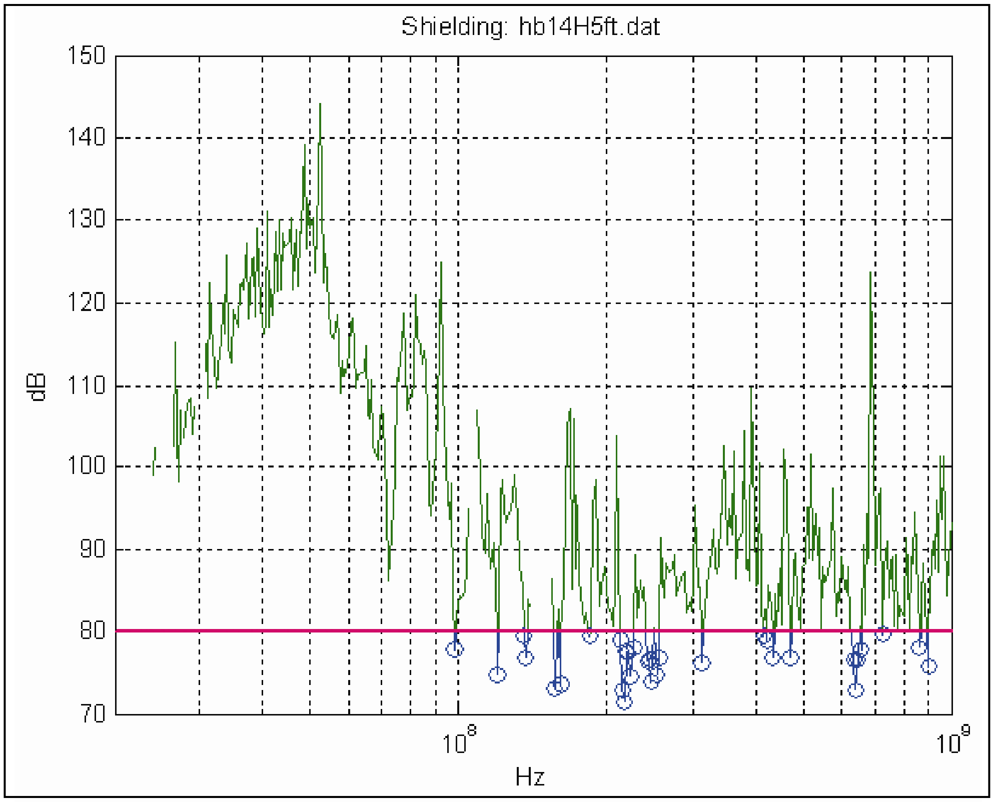

Figure A-7. Sample drop-out plot 1. (The points falling above the criteria threshold are plotted in green, and the points falling below the criteria threshold are plotted in blue with circle markers. The criteria threshold is shown in red. Data point, in which the dynamic range did not exceed 100 dB are not plotted.)

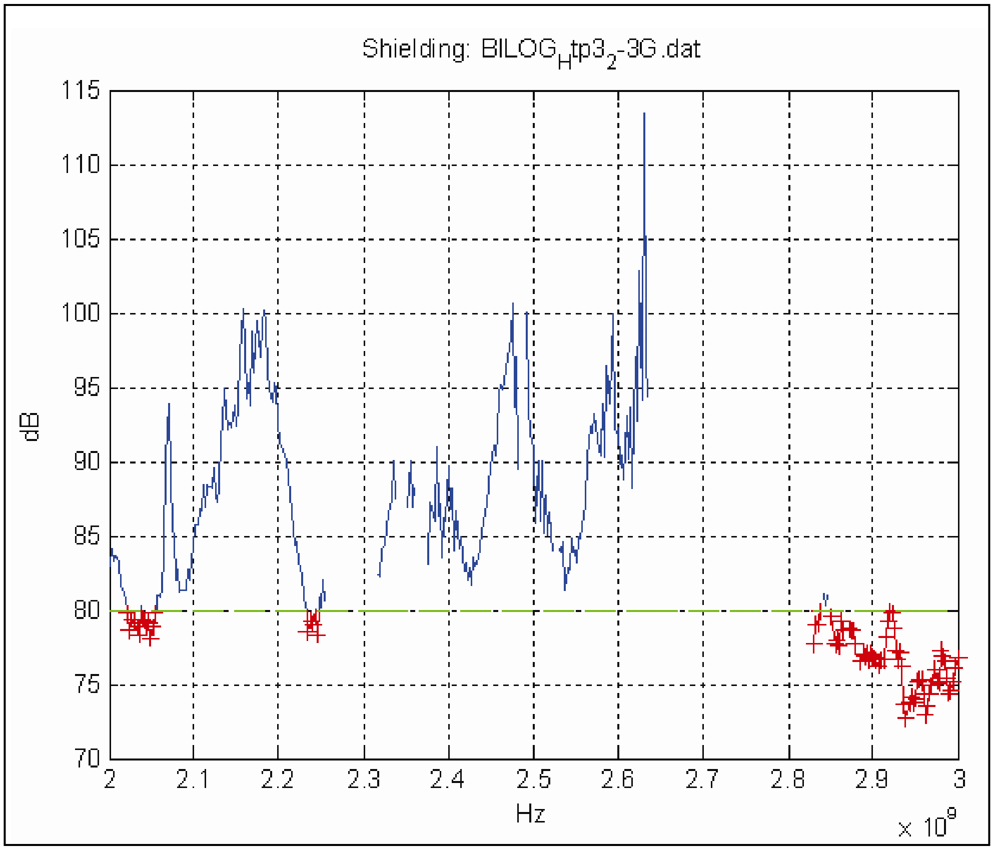

Figure A-8. Sample drop-out plot 2. (The points falling above the criteria threshold are plotted in blue, and the points falling below the criteria threshold are plotted in red with plus ("+") markers. The criteria threshold is shown in green. Data points in which the dynamic range did not exceed 100 dB are not plotted.)

INTENTIONALLY LEFT BLANK.

Appendix B. Partial Program Listing

```
function varargout = se4b(varargin)
% Apr,May 2004 Neal Tesny

% SE4B M-file for se4b.fig
%      SE4B, by itself, creates a new SE4B or raises the existing
%      singleton*.
%
%      H = SE4B returns the handle to a new SE4B or the handle to
%      the existing singleton*.
%
%      SE4B('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in SE4B.M with the given input arguments.
%
%      SE4B('Property','Value',...) creates a new SE4B or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before se4b_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to se4b_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help se4b

% Last Modified by GUIDE v2.5 18-Oct-2004 11:54:13

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @se4b_OpeningFcn, ...
    'gui_OutputFcn',  @se4b_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before se4b is made visible.
function se4b_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to se4b (see VARARGIN)

% Choose default command line output for se4b
```

```
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% initial variables:
handles.pname='C:\';
handles.fname='shlepped.dat';
%flags:
handles.refFlag=false;
handles.refattn1Flag=false;
handles.refattn2Flag=false;
handles.measFlag=false;
handles.measattn1Flag=false;
handles.noiseFlag=false;
handles.diffFlag=false;
% threshhold plot variables:
handles.colorBelow='r';
handles.colorAbove='b';
handles.threshLineType='--';
handles.threshLineColor='g';
handles.markerBelowType='+';
handles.markerAboveType=0;
handles.markerBelowColor='r';
handles.markerAboveColor='b';
handles.threshPlotGrid=true;
handles.threshPlotLog=true;
handles.threshLineWeight=0.5;
handles.lineWeightAbove=0.5;
handles.lineWeightBelow=0.5;
handles.ticLabelSize=10;
handles.axisLabelSize=10;
handles.titleSize=10;

%reg plot variables:
handles.diffLineType='-';
handles.diffLineColor='b';
handles.diffLineWidth=0.5;
handles.diffLineMarkerType='none';
handles.diffLineMarkerColor='b';
handles.diffLineMarkerSize=6;

% UIWAIT makes se4b wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = se4b_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
% --- Executes on button press in plotmeas.
function plotmeas_Callback(hObject, eventdata, handles)
% hObject    handle to plotmeas (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

…

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in loadref.
function loadref_Callback(hObject, eventdata, handles)
% hObject    handle to loadref (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

[filename2, pathname2] = uigetfile([pathname,filename],'Enter file name to
read:')
…

% --- Executes on button press in plotref.
function plotref_Callback(hObject, eventdata, handles)
% hObject    handle to plotref (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
…

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes on button press in plotdiff.
function plotdiff_Callback(hObject, eventdata, handles)
% hObject    handle to plotdiff (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% plot data…

% --- Executes on button press in recalc.
function recalc_Callback(hObject, eventdata, handles)
% hObject    handle to recalc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Recalculate.

% --- Executes on button press in loadmeas.
function loadmeas_Callback(hObject, eventdata, handles)
% hObject    handle to loadmeas (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in setspec.
function setspec_Callback(hObject, eventdata, handles)
% hObject    handle to setspec (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in checkboxSpec.
function checkboxSpec_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxSpec (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxSpec


% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in radiobutton1.
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton1


% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton2


% --- Executes on button press in checkbox2.
function checkbox2_Callback(hObject, eventdata, handles)
% hObject    handle to checkbox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkbox2


% --- Executes on button press in pushbutton12.
function pushbutton12_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton13.
function pushbutton13_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in loadrefattn1.
function loadrefattn1_Callback(hObject, eventdata, handles)
% hObject    handle to loadrefattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in plotrefattn1.
function plotrefattn1_Callback(hObject, eventdata, handles)
% hObject    handle to plotrefattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double


% --- Executes on button press in loadrefattn2.
function loadrefattn2_Callback(hObject, eventdata, handles)
% hObject    handle to loadrefattn2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in plotrefattn2.
```

```
function plotrefattn2_Callback(hObject, eventdata, handles)
% hObject    handle to plotrefattn2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as a
double


% --- Executes on button press in clearrefattn1.
function clearrefattn1_Callback(hObject, eventdata, handles)
% hObject    handle to clearrefattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in clearrefattn2.
function clearrefattn2_Callback(hObject, eventdata, handles)
% hObject    handle to clearrefattn2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in loadmeasattn1.
function loadmeasattn1_Callback(hObject, eventdata, handles)
% hObject    handle to loadmeasattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in plotmeasattn1.
function plotmeasattn1_Callback(hObject, eventdata, handles)
% hObject    handle to plotmeasattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in clearmeasattn1.
function clearmeasattn1_Callback(hObject, eventdata, handles)
% hObject    handle to clearmeasattn1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in loadnoise.
function loadnoise_Callback(hObject, eventdata, handles)
% hObject    handle to loadnoise (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in plotnoise.
function plotnoise_Callback(hObject, eventdata, handles)
% hObject    handle to plotnoise (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in clearnoise.
function clearnoise_Callback(hObject, eventdata, handles)
% hObject    handle to clearnoise (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in radiobutton3.
function radiobutton3_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton3


% --- Executes on button press in plotdynrange.
function plotdynrange_Callback(hObject, eventdata, handles)
% hObject    handle to plotdynrange (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in buttonThreshPlotOptions.
function buttonThreshPlotOptions_Callback(hObject, eventdata, handles)
% hObject    handle to buttonThreshPlotOptions (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function fileformat_CreateFcn(hObject, eventdata, handles)
% hObject    handle to fileformat (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: popupmenu controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes on selection change in fileformat.
function fileformat_Callback(hObject, eventdata, handles)
% hObject    handle to fileformat (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background, change
%       'usewhitebg' to 0 to use default.  See ISPC and COMPUTER.
usewhitebg = 1;
if usewhitebg
    set(hObject,'BackgroundColor',[.9 .9 .9]);
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of
slider


% --------------------------------------------------------------------
function Untitled_1_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Untitled_2_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
```

```
% --- Otherwise, executes on mouse press in 5 pixel border or over
buttonThreshPlotOptions.
function buttonThreshPlotOptions_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to buttonThreshPlotOptions (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in buttonplotThresh.
%function buttonplotThresh_Callback(hObject, eventdata, handles)
% hObject    handle to buttonplotThresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in buttonPlotThresh.
function buttonPlotThresh_Callback(hObject, eventdata, handles)
% hObject    handle to buttonPlotThresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in buttonRegPlotOptions.
function buttonRegPlotOptions_Callback(hObject, eventdata, handles)
% hObject    handle to buttonRegPlotOptions (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% out=100;
% while out~=0 & out~=10,


% --- Executes on button press in checkboxDiff.
function checkboxDiff_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxDiff (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxDiff


% --- Executes on button press in checkbox4.
function checkbox4_Callback(hObject, eventdata, handles)
% hObject    handle to checkbox4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkbox4


% --- Executes on button press in checkboxDyn.
function checkboxDyn_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxDyn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxDyn
```

```
% --- Executes on button press in checkboxSpec20.
function checkboxSpec20_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxSpec20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxSpec20


% --- Executes on button press in checkboxSpec2.
function checkboxSpec2_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxSpec2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxSpec2


% --- Executes on button press in buttonPlotReg.
function buttonPlotReg_Callback(hObject, eventdata, handles)
% hObject    handle to buttonPlotReg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in buttonDropPlotOptions.
function buttonDropPlotOptions_Callback(hObject, eventdata, handles)
% hObject    handle to buttonDropPlotOptions (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in buttonDropPlot.
function buttonDropPlot_Callback(hObject, eventdata, handles)
% hObject    handle to buttonDropPlot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes during object creation, after setting all properties.
function specAddOn_CreateFcn(hObject, eventdata, handles)
% hObject    handle to specAddOn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


function specAddOn_Callback(hObject, eventdata, handles)
% hObject    handle to specAddOn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in checkboxPlotAboveSpec.
function checkboxPlotAboveSpec_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxPlotAboveSpec (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbuttonHelp.
function pushbuttonHelp_Callback(hObject, eventdata, handles)
% hObject    handle to pushbuttonHelp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in buttonCloseFigs.
function buttonCloseFigs_Callback(hObject, eventdata, handles)
% hObject    handle to buttonCloseFigs (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in buttonSave.
function buttonSave_Callback(hObject, eventdata, handles)
% hObject    handle to buttonSave (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes during object deletion, before destroying properties.
function setspec_DeleteFcn(hObject, eventdata, handles)
% hObject    handle to setspec (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function specset_CreateFcn(hObject, eventdata, handles)
% hObject    handle to specset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes on selection change in specset.
function specset_Callback(hObject, eventdata, handles)
% hObject    handle to specset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

INTENTIONALLY LEFT BLANK.

Distribution

ADMNSTR
DEFNS TECHL INFO CTR
ATTN DTIC-OCP (ELECTRONIC COPY)
8725 JOHN J KINGMAN RD STE 0944
FT BELVOIR VA 22060-6218

DARPA
ATTN IXO S WELBY
3701 N FAIRFAX DR
ARLINGTON VA 22203-1714

OFC OF THE SECY OF DEFNS
ATTN ODDRE (R&AT)
THE PENTAGON
WASHINGTON DC 20301-3080

COMMANDER
21ST USA SIGNAL BRIGADE
ATTN CHIEF OF LOGISTICS
1435 PORTER STREET STE 100
FT DETRICK MD 21702-5046

US ARMY TRADOC
BATTLE LAB INTEGRATION & TECHL DIRCTRT
ATTN ATCD-B
10 WHISTLER LANE
FT MONROE VA 23651-5850

HQDA ODCS G-3/5/7
ATTN DAMO-SSD M F ALTFELD
400 ARMY PENTAGON
WASHINGTON DC 30210-0400

PEO EIS, PM DCATS
ATTN J JOO
FT MONMOUTH NJ 07703

PM DSCS TERMINALS, PM DCATS
ATTN AMSEL-IS-TSA-DSA
C BENJAMIN
ATTN AMSEL-IS-TSA-DSA V HANEY
ATTN SFAE-PS-TS-DSC
A B RICHMOND
ATTN SFAE-PS-TS-DSC M E BRYANT
ATTN SFAE-PS-TS-DSC R HYERS
ATTN AMSEL-IS-TSA-DSA
D SINGLETON
ATTN AMSEL-IS-TSA-DSA G
CHRISTOPHE
BLDG 209
FT MONMOUTH NJ 07703-5000

PM WIN-T
ATTN SFAE-C3T-WIN J R SHIELDS
BLDG 914
FT MONMOUTH NJ 07703

RRMC-ENO
ATTN M ALLEN
201 BEASLEY DR STE 100
FT DETRICK MD 21702-5029

SMC/GPA
2420 VELA WAY STE 1866
EL SEGUNDO CA 90245-4659

TECOM
ATTN AMSTE-CL
ABERDEEN PROVING GROUND MD
21005-5057

US ARMY ARDEC
ATTN AMSTA-AR-TD
BLDG 1
PICATINNY ARSENAL NJ 07806-5000

COMMANDING GENERAL
US ARMY AVN & MIS CMND
ATTN AMSAM-RD W C MCCORKLE
REDSTONE ARSENAL AL 35898-5000

US ARMY INFO SYS ENGRG CMND
ATTN AMSEL-IE-TD F JENIA
FT HUACHUCA AZ 85613-5300

US ARMY NATICK RDEC ACTING
TECHL DIR
ATTN SBCN-TP P BRANDLER
KANSAS STREET BLDG 78
NATICK MA 01760-5056

US ARMY NETWORK ENTERPRISE
TECHNLGY CMND
9TH ARMY SIGNAL COMMAND
ATTN G4, OFFICE OF THE COMMAND
ENGINEER D SCHOW
2133 CUSHING STREET SUITE 2313
FT HUACHUCA AZ 85613-7070

US ARMY NETWORK ENTERPRISE
TECHNLGY CMND
9TH ARMY SIGNAL COMMAND
ATTN O WITT
2133 CUSHING STREET BLDG 61801
RM 3105
FT HUACHUCA AZ 85613-7070

US ARMY SIMULATION TRAIN &
INSTRMNTN CMND
ATTN AMSTI-CG M MACEDONIA
12350 RESEARCH PARKWAY
ORLANDO FL 32826-3726

MCS (HQ)
ATTN GIGSG/KCP J THORNE
HANSCOM AFB MA 01731-1620

USSTRATCOMCL113
ATTN F J METZGER JR
901 SAC BLVD STE M102
OFFUTT AFB NE 68113-6300

FEMMECOMP INC
ATTN R LOUNSBERY
14170 NEWBROOK DR
CHANTILLY VA 20151

J D ENGEERING
ATTN J DANDO (10 COPIES)
3103 FOX DEN LN
OAKTON VA 22124

US ARMY RSRCH LAB
ATTN AMSRD-ARL-CI-OK-TP TECHL
LIB T LANDFRIED (2 HC)
ABERDEEN PROVING GROUND MD
21005-5066

DIRECTOR
US ARMY RSRCH LAB
ATTN AMSRD-ARL-RO-EN W D BACH
PO BOX 12211
RESEARCH TRIANGLE PARK NC 27709

US ARMY RSRCH LAB
ATTN AMSRD-ARL-CI-OK-T TECHL
PUB (2 COPIES)
ATTN AMSRD-ARL-CI-OK-TL TECHL
LIB (2 COPIES)
ATTN AMSRD-ARL-D J M MILLER
ATTN AMSRD-ARL-D J ROCCHIO
ATTN AMSRD-ARL-SE-D E SHAFFER
ATTN AMSRD-ARL-SE-DE D CONRAD
(3 COPIES)
ATTN AMSRD-ARL-SE-DE L DILKS
(3 COPIES)
ATTN AMSRD-ARL-SE-DE N TESNY
(20 COPIES)
ATTN AMSRD-ARL-SE-DE R
ATKINSON (3 COPIES)
ATTN AMSRD-ARL-SE-DP S BAYNE
ATTN AMSRL-SE-DS B KING
(3 COPIES)
ATTN IMNE-ALC-IMS MAIL &
RECORDS MGMT
ADELPHI MD 20783-1197